

## Program Analysis Project

The tool I decided on using for this project was Cppcheck. It is a static analysis tool that for C and C++ code. Cppcheck detects bugs and undefined behavior and dangerous coding constructs. Its goal is to limit as many false positives as possible. The tool is designed to analyze C/C++ code even when the code is not in its standard syntax.

The software I chose to run Cppcheck on was source C code files for BACnet Protocol Stack library, which provides an application layer, network layer, and MAC layer communication services for multiple platforms. The source code for this library is written in C and designed to be portable across many compilers and architectures, specifically designed for use in embedded BACnet devices. The specific library was version 0.8.6 which contains a total of 62 source C files. This experiment was executed on a Windows 10 Pro platform on an AMD Ryzen 7 2700X 3.70GHz processor and 16GB RAM. In the experiment, the entire source library and their libraries were scanned for errors, warnings, style warnings, portability, performance warnings, and information.

From the scans, it was found that there were 34 warnings and 8 errors that span across 11 of the 62 source C files. Of the 8 errors, 2 were from one file (bvlc.c) and the last 6 were present in another file (bacint.c). In the bvlc.c file, there were two instances where there were potential Array out of bounds errors, where an index table is being accessed through a for loop and that index passes the bounds of the table. In the bacint.c file, the 6 instances of errors are all signed integer overflow errors for an expression. The expression is the same for each error, a variable that's pointer value is being accessed and changed. This test takes no longer than 11 seconds to execute. The static analysis software was not difficult to use at all.

For the array out of bounds error, I believe that it is being caused because the last value of the incremented index is being used by the analysis tool to assume that it will access this last index, where the last index is actually only used to exit the loop. That is my guess but, I cannot confirm it without dynamically running and monitoring the library itself. And given that it is a library, some code that uses said library must be developed that accesses the file. For the signed integer overflow error, I am not able to verify whether these are false positives or not. More analysis of the code itself would have to be researched before I can determine if it's a true positive error.

Cppcheck - Project: bacapp.c

File Edit View Analyze Help

Quick Filter:

File	Severity	Line	Summary	Since date	Tag
> apdu.c					
> arf.c					
> awf.c					
> bacaddr.c					
> bacapp.c					
> bacdcode.c					
> bacdevobjpro...					
> bacerror.c					
▼ bacint.c					
> bacint.c	error	136	Signed integer overflow for expression '%value'.		
> bacint.c	error	162	Signed integer overflow for expression '%value'.		
> bacint.c	error	163	Signed integer overflow for expression '%value'.		
> bacint.c	error	190	Signed integer overflow for expression '%value'.		
> bacint.c	error	191	Signed integer overflow for expression '%value'.		
> bacint.c	error	192	Signed integer overflow for expression '%value'.		
bacint.c	information	239	--check-library: There is no matching configuration ...		
bacint.c	information	240	--check-library: There is no matching configuration ...		
bacint.c	information	241	--check-library: There is no matching configuration ...		
bacint.c	information	242	--check-library: There is no matching configuration ...		

Array 'BBMD\_Table[128]' accessed at index 128, which is out of bounds.

```
1805     BBMD_Table[i].dest_port == entry->dest_port) {
1806         return false;
1807     }
1808 }
1809
1810 if(!found)
1811     return false;
1812
1813 /* Copy new entry to the empty slot */
1814 BBMD_Table[i] = *entry;
1815 BBMD_Table[i].valid = true;
1816 /* BDT changed! Save backup to file */
1817 bvlc_bdt_backup_local();
1818
1819 return true;
1820 }
1821
1822 /** Enable NAT handling and set the global IP address
1823  * @param [in] - Global IP address visible to peer BBMDs and foreign devices
1824  */
1825 void bvlc_set_global_address_for_nat(const struct in_addr* addr)
1826 {
1827     BVLC_Global_Address = *addr;
1828     BVLC_NAT_Handling = true;
1829 }
1830
1831 /** Disable NAT handling.
1832  */
1833 void bvlc_disable_nat(void)
```

Analysis Log Warning Details

Cppcheck - Project: bacapp.c

File Edit View Analyze Help

Quick Filter:

File	Severity	Line	Summary	Since
bvlc.c	style	717	The scope of the variable 'BVLC_length' can be reduced.	
bvlc.c	style	344	Variable 'BVLC_length' is assigned a value that is never used.	
bvlc.c	style	680	Variable 'i' is assigned a value that is never used.	
bvlc.c	style	681	Variable 'BVLC_length' is assigned a value that is never used.	
bvlc.c	style	716	Variable 'i' is assigned a value that is never used.	
bvlc.c	style	717	Variable 'BVLC_length' is assigned a value that is never used.	
bvlc.c	error	1814	Array 'BBMD_Table[128]' accessed at index 128, which is out of bounds.	
bvlc.c	error	1815	Array 'BBMD_Table[128]' accessed at index 128, which is out of bounds.	
bvlc.c	information	297	--check-library: There is no matching configuration for function bvlc_e...	
bvlc.c	information	435	--check-library: There is no matching configuration for function encod...	
bvlc.c	information	463	--check-library: There is no matching configuration for function bvlc_e...	

Array 'BBMD\_Table[128]' accessed at index 128, which is out of bounds.

```
1801
1802     /* Make sure that we are not adding a duplicate */
1803     if(BBMD_Table[i].dest_address.s_addr == entry->dest_address.s_addr &&
1804        BBMD_Table[i].broadcast_mask.s_addr == entry->broadcast_mask.s_addr &&
1805        BBMD_Table[i].dest_port == entry->dest_port) {
1806         return false;
1807     }
1808 }
1809
1810 if(!found)
1811     return false;
1812
1813 /* Copy new entry to the empty slot */
1814 BBMD_Table[i] = *entry;
1815 BBMD_Table[i].valid = true;
1816 /* BDT changed! Save backup to file */
1817 bvlc_bdt_backup_local();
1818
1819 return true;
1820 }
1821
1822 /** Enable NAT handling and set the global IP address
1823  * @param [in] - Global IP address visible to peer BBMDs and foreign devices
1824  */
1825 void bvlc_set_global_address_for_nat(const struct in_addr* addr)
1826 {
1827     BVLC_Global_Address = *addr;
1828     BVLC_NAT_Handling = true;
1829 }
1830
1831 /** Disable NAT handling.
1832  */
1833 void bvlc_disable_nat(void)
1834 {
1835     BVLC_NAT_Handling = false;
1836     BVLC_Global_Address.s_addr = 0;
1837 }
1838
```

Analysis Log Warning Details